In the Specification:

On page 4, please replace the first paragraph starting on line 1 with the following rewritten paragraph:

A1

Therefore, a need exists for an executable program, such as a software driver, that drives a plurality of peripherals and can run more efficiently on a plurality of [CPUS's] CPUs and host systems.

On page 4, please replace the sixth paragraph starting on line 15 with the following rewritten paragraph:

A2

FIG. 4 is a flow chart illustrating one example [of A method] of a method for constructing an executable program in memory in accordance with one embodiment of the invention.

On page 4, please replace the eighth paragraph starting on line 19 with the following rewritten paragraph:

A3

FIG. 6 is a graphic illustration [of A mini-driver] of a mini-driver and jump instruction patching in accordance with one embodiment of the invention.

On page 4, please replace the tenth paragraph, starting at line 30 and continuing to page 5, with the following rewritten paragraph:

A4

A method and apparatus for constructing an executable program, such as drivers, in memory includes obtaining actual dynamic and static system configuration parameters and dynamically constructing driver code bundles from a set of code modules obtained from a library, based on the actual system configuration parameters. The set of code modules includes code modules associated with a plurality of system configuration parameters. One example of the actual system configuration parameters includes static system configuration parameters such

as in the case of a computer, a CPU type, clock speed and [system memory size graphic accelerator type] system memory size, graphic accelerator type and associated video memory size. Other actual system configuration parameters include dynamic configuration parameters which can be changed by the user or other application. One example of a dynamic configuration parameter may be, for example, pixel depth and display screen resolution. After obtaining optimal system configuration parameters depending upon a system's setting or configurations, dedicated code fragments such as code modules are used and stored in system memory or other suitable memory. Accordingly, optimal driver code is loaded at all times for a particular chip set and no unnecessary code is loaded from a CD ROM or other source. Since the code modules are selected based on a given set of configuration parameters an optimal driver is stored for a given system. The two types of actual system configuration parameters that determine a dynamic code bundle are the static, such as configuration parameters that never change during an operating session (hardware configuration) and dynamic configuration parameters that depend, for example, on software configuration (system flags, pixel depth, screen resolution, etc.). The dynamic configuration parameters and the static system configuration parameters are stored and are used as a type of index to determine a set of code modules that are selected to define a code bundle. A different code bundle may be stored in memory depending upon the different dynamic configuration parameter being selected.

On page 7, please replace the first paragraph with the following rewritten paragraph:

The mini-driver 100 may be a 16-bit .dll or any other suitable size or type of software code. Each and every entry point 102a-102n associated with a given driver (i.e., functions of the driver) are [patched with by a jump] patched with a jump instruction. Accordingly, the mini-driver 100 includes a jump instruction for each entry point of a display driver, for example.